UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/620,748	07/16/2003	Mark S. Moir	6000-33800	8974

58467 7590 10/16/2008
MHKKG/SUN
P.O. BOX 398
AUSTIN, TX 78767

EXAMINER	
TSAI, SHENG JEN	

ART UNIT	PAPER NUMBER
2186	

MAIL DATE	DELIVERY MODE
10/16/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 10/620,748	**Applicant(s)** MOIR ET AL.	
	Examiner SHENG-JEN TSAI	**Art Unit** 2186	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _05 August 2008_.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-44_ is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _1-44_ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on _16 July 2003_ is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This Office Action is taken in response to Applicant's Amendments and Remarks filed on August 5, 2008 regarding application 10/620,748 filed on July 16, 2003.

2. Claims 1-30, 36, 38-40 and 43 have been amended.

Claims 1-44 are pending under consideration.

3. ***Response to Amendments and Remarks***

Applicants' remarks have been fully and carefully considered with Examiner's response set forth below.

(1) In view of the amendments, rejection of claim 18 under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter has been withdrawn.

(2) In view of the amendments, rejection of claim 20 under 35 U.S.C. 112, 2nd paragraph has been withdrawn.

(3) Applicants amend independent claims with additional limitations of "mediating concurrent execution of the access operations using a single-target synchronization primitive" and "wherein the single-target of the single-target synchronization primitive includes a value encoding or an element of the array and a version number encoded integrally therewith," and contend that the Martin reference fails to teach these amended limitations. The Examiner disagrees.

First, Martin teaches concurrent execution of the access operations [The present invention relates generally to coordination amongst execution sequences in a multiprocessor computer, and more particularly, to structures and techniques for facilitating non-blocking access to concurrent shared objects (paragraph 0004); Sometimes an implementation of such a data structure is shared among multiple concurrent processes … (paragraph 0007)].

Second, Martin teaches the compare-and-swap (CAS) operation [paragraph 0012] and that CAS operations are single-target synchronization primitives [existing synchronization operations on single memory locations, such as compare-and-swap (CAS) … (paragraph 0013)].

Therefore, Martin clearly teaches the limitation of “mediating concurrent execution of the access operations using a single-target synchronization primitive.”

Third, Martin teaches [The "compare-and-swap" operation (CAS) typically accepts three values or quantities: a memory address A, a comparison value C, and a new value N. The operation fetches and examines the contents V of memory at address A. If those contents V are equal to C, then N is stored into the memory location at address A, replacing V. Whether or not V matches C, V is returned or saved in a register for later inspection. All this is implemented in a linearizable, if not atomic, fashion. Such an operation may be notated as "CAS(A, C, N)" (paragraph 0012)].

Fourth, figure 1 of Martin shows that elements of the array has a V number, such as V1, V2, V3 and son on.

Therefore, Martin clearly teaches the limitation of “wherein the single-target of the single-target synchronization primitive includes a value encoding or an element of the array and a version number encoded integrally therewith.”

Another iteration of claim analysis based on the Martin references addressing all previously presented and currently amended limitations has been made. Refer to the corresponding sections of the following claim analysis for details.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

5. Claims 1-4, 6-10, 12-14, 16-22, 24, 26-33 and 35-44 are rejected under 35 U.S.C. 102(a), as well as under 35 U.S.C. 102(e), as being anticipated by Martin et al. (US Patent Application Publication 2001/0047361, hereinafter referred to as Martin).

As to claim 1, Martin discloses **a computer readable storage medium encoding program code** [paragraphs 0048, 0084, 0087, 0093, 0096] **executable on one or more processors** [The present invention relates generally to coordination amongst execution sequences in a multiprocessor computer, and more particularly, to structures and techniques for facilitating non-blocking access to concurrent shared objects (paragraph 0004)] **to implement: instantiating a data structure implementation in a memory** [A computer program product encoded in at least one computer readable medium, the computer program product comprising: … (Claim 57)], **comprising**:

a double-ended array [as shown in figure 1; An important abstract data structure in computer science is the "double-ended queue" (abbreviated "deque" and pronounced "deck"), which is a linear sequence of items, usually initially empty, that supports the four operations of inserting an item at the left-hand end ("left push"), removing an item from the left-hand end ("left pop"), inserting an item at the right-hand end ("right push"), and removing an item from the right-hand end ("right pop") (paragraph 0006)];

executing a plurality of opposing-end access operations [the corresponding “opposite-end” comprises the “left hat” (figure 1, 103) and the “right hat “ (figure 1, 104)] **that, when executed on the one or more processors, access the memory, and provide concurrent push-type and pop-type access to at least one of the opposing ends and concurrent** [as shown in figure 1; An important abstract data structure in computer science is the "double-ended queue" (abbreviated "deque" and pronounced "deck"), which is a linear sequence of items, usually initially empty, that supports the four operations of inserting an item at the left-hand end ("left push"), removing an item from the left-hand end ("left pop"), inserting an item at the right-hand end ("right push"), and removing an item from the right-hand end ("right pop") (paragraph 0006)], **opposing-end accesses that are non-interfering for at least some states of the array** [as shown in figure 1, the left hat and the right hat are operating on different elements located at the opposite ends, thus non-interfering with each other]**, and**

mediating concurrent execution of the access operations using a single-target synchronization primitive [The present invention relates generally to coordination amongst execution sequences in a multiprocessor computer, and more particularly, to structures and techniques for facilitating non-blocking access to concurrent shared objects (paragraph 0004); Sometimes an implementation of such a data structure is shared among multiple concurrent processes … (paragraph 0007); Martin teaches the compare-and-swap (CAS) operation (paragraph 0012) and that CAS operations are single-target synchronization primitives (existing synchronization operations on single memory locations, such as compare-and-swap (CAS) … (paragraph 0013)]; **wherein the data structure implementation is linearizable and non-blocking** [A set of structures and techniques are described herein whereby an exemplary concurrent shared object, namely a double-ended queue (deque), is implemented. Although <u>non-blocking, linearizable deque implementations exemplify several advantages of realizations in accordance with the present invention</u>, the present invention is not limited thereto (paragraph 0018)], **and wherein the single-target of the single-target synchronization primitive includes a value encoding or an element of the array and a version number encoded integrally therewith** [The "compare-and-swap" operation (CAS) typically accepts three values or quantities: a memory address A, a comparison value C, and a new value N. The operation fetches and examines the contents V of memory at address A. If those contents V are equal to C, then N is stored into the memory location at address A, replacing V. Whether or not V matches C, V is returned or saved in a register for later inspection. All this is implemented in a linearizable, if not atomic, fashion. Such an operation may be notated as "CAS(A, C, N)" (paragraph 0012); Fourth, figure 1 of Martin shows that elements of the array has a V number, such as V1, V2, V3 and son on].

As to claim 2, Martin teaches that **the storage medium of claim 1, wherein the concurrent opposing-end access operations are non-interfering for all but boundary condition states of the array** [We begin by describing the operation of "normal" push and pop operations that do not encounter any boundary cases or concurrent operations. Later, we describe special cases for these operations, interaction with concurrent operations, and operations for growing and shrinking the list (paragraph 0104)].

As to claim 3, Martin teaches that **the storage medium of claim 1, wherein the non-blocking implementation is obstruction-free, though not wait-free or lock-free** [as shown in figure 1, the left hat and the right hat are operating on different elements located at the opposite ends, thus obstruction-free; and when both of them try to access the same element, then only one may access and synchronization is needed, hence not wait-free or lock-free].

As to claim 4, Martin teaches that **the storage medium of claim 1, wherein the single-target synchronization primitive employs a Compare-And-Swap (CAS) operation** [In other realizations, a synchronization primitive such as a CAS can be used to ensure a precise count (paragraph 0123); we first use a CAS primitive to set the next node's value field to the distinguishing value RY (lines 5-7). The reason for using a CAS instead of an ordinary store is that we can determine the value overwritten when storing the RY value (paragraph 0141)].

As to claim 6, Martin teaches that **the storage medium of claim 4, wherein said mediating comprises attempting to increment the version number included in the single-target of the single-target synchronization primitive includes a value encoding for an element of the array and a version number encoded integrally therewith** [The "compare-and-swap" operation (CAS) typically accepts three values or quantities: a memory address A, a comparison value C, and a new value N (paragraph 0012); figure 1 of Martin shows that elements of the array has a V number, such as V1, V2, V3 and son on; Furthermore, although various non-blocking, linearizable deque implementations described herein <u>employ a particular synchronization primitive, namely a double compare and swap (DCAS) operation</u>, the present invention is not limited to DCAS-based realizations (paragraph 0019); In other realizations, <u>a synchronization primitive</u> such as a <u>CAS</u> can be used to ensure a precise count (paragraph 0123); we first <u>use a CAS primitive</u> to set the next node's value field to the distinguishing value RY (lines 5-7). The reason for using a CAS instead of an ordinary store is that we can determine the value overwritten when storing the RY value (paragraph 0141)].

As to claim 7, Martin teaches that **the storage medium of claim 1, wherein the double-ended array implements a deque** [An important abstract data structure in computer science is the "<u>double-ended queue</u>" (abbreviated "<u>deque</u>" and pronounced "deck") … (paragraph 0006)].

As to claim 8, Martin teaches that **the storage medium of claim 1, wherein the opposing-end access operations are at least consistent with semantics of a FIFO**

queue [Many variations, modifications, additions, and improvements are possible. For example, while various full-function deque realizations have been described in detail, realizations of other shared object data structures, including realizations that forgo some of access operations, e.g., for use as a FIFO, queue, LIFO, stack or hybrid structure, will also be appreciated by persons of ordinary skill in the art (paragraph 0153)].

As to claim 9, Martin teaches that **the storage medium of claim 1, wherein the boundary-condition states include an empty state** [Except in a special case, which is described later, two shared pointers, hereafter RHat and LHat, point to the right and left sentinels, respectively. For an empty state of the deque, left and right sentinels are adjacent. Thus, FIG. 10A depicts one representation of an empty deque (paragraph 0100)].

As to claim 10, Martin teaches that **the storage medium of claim 1, wherein the boundary-condition states include a single element state** [A DCAS failure means that either the hat has been moved by another push or pop at the same end of the queue or (in the case of a single element deque) the targeted node was popped from the opposing end of the deque. In either case, the pop_right operation loops for another attempt (paragraph 0088)].

As to claim 12, Martin teaches that **the storage medium of claim 11, wherein the boundary-condition states include a full state** [The deque is initially in the empty state (following invocation of make_deque( )), that is, has cardinality 0, and is said to have reached a full state if its cardinality is max_length_S. In general, for deque implementations described herein, cardinality is unbounded except by limitations (if any) of an underlying storage allocator (paragraph 0054)].

As to claim 13, Martin teaches that **the storage medium of claim 11, wherein the opposing-end accesses include opposing-end, push-type accesses; and wherein the boundary-condition states include a nearly full state** [An important abstract data structure in computer science is the "double-ended queue" (abbreviated "deque" and pronounced "deck"), which is a linear sequence of items, usually initially empty, that supports the four operations of inserting an item at the left-hand end ("left push"), removing an item from the left-hand end ("left pop"), inserting an item at the right-hand end ("right push"), and removing an item from the right-hand end ("right pop") (paragraph 0006); Ideally, operations on one end of the deque would never impede operations on the other end of the deque unless the deque were nearly empty (containing two items or fewer) or, in some implementations, nearly full (paragraph 0009)].

As to claim 14, Martin teaches that **the storage medium of claim 1, wherein distinct left null and right null distinguishing values are employed to identify free elements of the array** [as shown in figure 1, the left hat and the right hat are operating on different elements located at the opposite ends, thus indicating free elements; if both of them are pointing to the same element, then it is an indication that there is no free element].

As to claim 16, Martin teaches that **the storage medium of claim 1, embodied as a software component combinable with program code to provide the program**

code with non-blocking access to a concurrent shared object [A concurrent shared object representation encoded in one or more computer readable media, the concurrent shared object representation comprising: … (Claim 56)].

As to claim 17, Martin teaches that **the storage medium of claim 1, embodied as a program executable to provide non-blocking access to a concurrent shared object** [Concurrent Shared Object Implemented Using a Linked-List with Amortized Node Allocation (title); Although non-blocking, linearizable deque implementations exemplify several advantages of realizations in accordance with the present invention, the present invention is not limited thereto (paragraph 0018); Claims 56-57].

As to claim 18, Martin teaches **the storage medium of claim 1, comprising at least one medium selected from the set of a disk, tape or other magnetic, optical, or electronic storage medium** [The computer program product of 57, wherein the at least one computer readable medium is selected from the set of a disk, tape or other magnetic, optical, or electronic storage medium and a network, wireline, wireless or other communications medium (Claim 62)].

As to claim 19, it recites substantially the same limitations as in claim 1, and is rejected for the same reasons set forth in the analysis of claim 1. Refer to "As to claim 1" presented earlier in this Office Action for details.

As to claim 20, it recites substantially the same limitations as in claim 3, and is rejected for the same reasons set forth in the analysis of claim 3. Refer to "As to claim 3" presented earlier in this Office Action for details.

As to claim 21, it recites substantially the same limitations as in claim 2, and is rejected for the same reasons set forth in the analysis of claim 2. Refer to "As to claim 2" presented earlier in this Office Action for details.

As to claim 22, it recites substantially the same limitations as in claim 6, and is rejected for the same reasons set forth in the analysis of claim 6. Refer to "As to claim 6" presented earlier in this Office Action for details.

As to claim 24, it recites substantially the same limitations as in claim 4, and is rejected for the same reasons set forth in the analysis of claim 4. Refer to "As to claim 4" presented earlier in this Office Action for details.

As to claim 26, Martin teaches **the storage medium of claim 19, wherein at least some concurrently executed access operations interfere with each other; and wherein the interfering concurrently executed access operations are each retried** [as shown in figure 1, the left hat and the right hat are operating on different elements located at the opposite ends, thus obstruction-free; and when both of them try to access the same element, then only one may access and synchronization is needed; If one or more other executions of push_right operations intervene and consume the newly allocated nodes, <u>this retry behavior</u> will again note the shortage and again call upon add_right_nodes to allocate more nodes until eventually there is at least one (paragraph 0085)].

As to claim 27, Martin teaches **the storage medium of claim 26, wherein the non-blocking deque implementation does not guarantee that at least one of the interfering concurrently executed access operations makes progress** [as shown in figure 1, the left hat and the right hat are operating on different elements located at the opposite ends, thus obstruction-free; and when both of them try to access the same element, then synchronization is needed, hence not wait-free or lock-free].

As to claim 28, Martin teaches **the storage medium of claim 27, wherein a separate contention management facility is employed to ensure progress in a concurrent computation that employs the deque implementation** [Concurrent Shared Object Implemented Using a Linked-List with Amortized Node Allocation (title); Furthermore, although various non-blocking, linearizable deque implementations described herein employ a particular synchronization primitive, namely a double compare and swap (DCAS) operation, the present invention is not limited to DCAS-based realizations (paragraph 0019); paragraph 0108].

As to claim 29, refer it recites substantially the same limitations as in claim 1, and is rejected for the same reasons set forth in the analysis of claim 1. Refer to “As to claim 1” presented earlier in this Office Action for details.

As to claim 30, it recites substantially the same limitations as in claim 1, and is rejected for the same reasons set forth in the analysis of claim 1. Refer to “As to claim 1” presented earlier in this Office Action for details.

As to claim 31, it recites substantially the same limitations as in claim 2, and is rejected for the same reasons set forth in the analysis of claim 2. Refer to “As to claim 2” presented earlier in this Office Action for details.

As to claim 32, it recites substantially the same limitations as in claim 3, and is rejected for the same reasons set forth in the analysis of claim 3. Refer to “As to claim 3” presented earlier in this Office Action for details.

As to claim 33, it recites substantially the same limitations as in claim 4, and is rejected for the same reasons set forth in the analysis of claim 4. Refer to “As to claim 4” presented earlier in this Office Action for details.

As to claim 35, it recites substantially the same limitations as in claim 1, and is rejected for the same reasons set forth in the analysis of claim 1. Refer to “As to claim 1” presented earlier in this Office Action for details.

As to claim 36, it recites substantially the same limitations as in claim 28, and is rejected for the same reasons set forth in the analysis of claim 28. Refer to “As to claim 28” presented earlier in this Office Action for details.

As to claim 37, it recites substantially the same limitations as in claim 28, and is rejected for the same reasons set forth in the analysis of claim 28. Refer to “As to claim 28” presented earlier in this Office Action for details. Further, figures 1-14 of Martin illustrate the changing occurring the operations.

As to claim 38, it recites substantially the same limitations as in claim 28, and is rejected for the same reasons set forth in the analysis of claim 28. Refer to “As to claim 28” presented earlier in this Office Action for details.

As to claim 39, refer it recites substantially the same limitations as in claim 28, and is rejected for the same reasons set forth in the analysis of claim 28. Refer to “As to claim 28” presented earlier in this Office Action for details.

As to claim 40, it recites substantially the same limitations as in claim 1, and is rejected for the same reasons set forth in the analysis of claim 1. Refer to “As to claim 1” presented earlier in this Office Action for details.

As to claim 41, it recites substantially the same limitations as in claim 27, and is rejected for the same reasons set forth in the analysis of claim 27. Refer to "As to claim 27" presented earlier in this Office Action for details.

As to claim 42, it recites substantially the same limitations as in claim 28, and is rejected for the same reasons set forth in the analysis of claim 28. Refer to "As to claim 28" presented earlier in this Office Action for details.

As to claim 43, it recites substantially the same limitations as in claim 1, and is rejected for the same reasons set forth in the analysis of claim 1. Refer to "As to claim 1" presented earlier in this Office Action for details.

As to claim 44, it recites substantially the same limitations as in claim 28, and is rejected for the same reasons set forth in the analysis of claim 28. Refer to "As to claim 28" presented earlier in this Office Action for details.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 5, 25 and 34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Martin et al. (US Patent Application Publication 2001/0047361, hereinafter referred to as Martin), and in view of Rowlands (US Patent Application Publication 2003/0217115).

Regarding claims 5, 25 and 34, Martin does not teach the single-target synchronization primitive employs a Load-Linked (LL) and Store-Conditional (SC) operation pair.

However, Rowlands teaches this limitation in the invention "Load-Linked/Store Conditional Mechanism in a CC-NUMA System," where the LL/SC operation pair is used for synchronization in a multiprocessor system [paragraphs 0003-0008].

Rowlands also teaches that the motivation of using LL/SC as a synchronization mechanism is because it allows other processors to access the memory location for which the atomic read-modify-write is being attempted, which is not permitted in other synchronization mechanisms such as atomic read-modify-write [With the load-linked/store conditional mechanism, other processors may access the memory location for which the atomic read-modify-write is being attempted. If a modification occurs, the load-linked/store conditional sequence is repeated. When the store conditional completes successfully, an atomic read-modify-write of the location has been performed (paragraph 0008)].

Therefore, it would have been obvious for one of ordinary skills in the art at the time of Applicants' invention to use LL/SC as a synchronization mechanism, as demonstrated by Rowlands, and to incorporate it into the existing scheme disclosed by Martin, in order to allows other processors to access the memory location for which the atomic read-modify-write is being attempted, which enhances the overall throughput of the system.

7. Claims 11, 15 and 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Martin et al. (US Patent Application Publication 2001/0047361, hereinafter referred to as Martin), and in view of Latour (US Patent Application Publication 2002/0078123).

Regarding claims 11, 15 and 23, Martin does not teach the array is indexable as a circular array.

However, Latour teaches this limitation in the invention "Method and Apparatus for Resource Access Synchronization" [The sequence of mutexes can be held in an array, a ring buffer, a linked list, <u>a circular linked list</u>, or any other suitable data structure that enables the order to be preserved (paragraph 0024); FIG. 9 illustrates the storage approach used in preferred embodiment of the invention. This uses a <u>circular linked list</u>, which is a combination of a bi-directional linked list and a ring buffer (paragraph 0054)].

Latour also teaches that the motivation of using a circular linked list is because it provides more flexibility than other types of storages [The use of the circular linked list buffer provides flexibility in that the size of the ring can readily be changed. In this manner, the number of storage locations in the ring can readily be changed to take account of changing circumstances, while still exercising control over the memory required successfully, an atomic read-modify-write of the location has been performed (paragraph 0054)].

Therefore, it would have been obvious for one of ordinary skills in the art at the time of Applicants' invention to use a circular linked list, as demonstrated by Latour, and to incorporate it into the existing scheme disclosed by Martin, in order to provide more flexibility.

8. ***Related Prior Art of Record***

The following list of prior art is considered to be pertinent to applicant's invention, but not relied upon for claim analysis conducted above.

- Shavit et al., (US 7,000,234), "Maintaining a Double-Ended Queue as a Linked-List with Sentinel Nodes and Delete Flags with Concurrent Non-Blocking Insert and Remove Operations Using a Double Compare-and-Swap Primitive."
- Steele, Jr. et al., (US Patent Application Publication 2001/0056420), "Lock-Free Implementation of Concurrent Shared Object with Dynamic Node Allocation and Distinguishing Pointer Value."
- Hu, (US 6,615,216), "Lock Free Data Structure Maintenance."
- Martin et al., (US Patent Application Publication 2006/0161737), "Concurrent Technique for Shared Objects."
- Harris (US 7,117,502), "Linked-List Implementation of a Data Structure with Concurrent Non-Blocking Insert and Remove Operations."
- Bonola (US 6,173,373), "Method and Apparatus for Implementing Stable Priority Queues Using Concurrent Non-Blocking Queuing Techniques."

Conclusion

9. Claims 1-44 are rejected as explained above.

10. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).
A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within

TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Sheng-Jen Tsai whose telephone number is 571-272-4244. The examiner can normally be reached on 8:30 - 5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Matthew Kim can be reached on 571-272-4182. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Sheng-Jen Tsai/

TFSA Examiner, Art Unit 2186

October 12, 2008